A MoBiMouse Plus szótárak készítése közben szerzett tapasztalatokról

(Egy szótárfeldolgozó programcsomag elvi lehetőségei, emberi segítséggel)

Vörösss Ferenc, Trepák Mónika

MorphoLogic
1126 Budapest Orbánhegyi út 5.
{voross,trepak}@morphologic.hu

Kivonat: Dolgozatunkban tipografizált szótári adatbázisok jelölőnyelvi adatbázissá alakítására tett, különböző feldolgozási elméletekhez kapcsolódó kísérleteiről szólunk. Felvázoljuk, hogy a szótári adatbázisok kialakítása, illetve az ezek intelligens keresését lehetővé tevő munkafolyamatok során milyen tapasztalatokkal gazdagodtunk a szótárírási szokásokat illetően. Cikkünk tanácsokkal és javaslatokkal szeretné segíteni azokat a lexikográfusokat, akik szeretnék szótáraikat egyszerűbben feldolgozható, átláthatóbb, egységesebb, felhasználóbarátabb szerkezetben szerkeszteni, és nem szeretnének az általunk tapasztaltakhoz hasonló hibákat elkövetni. Javaslatainkat példákkal is illusztráljuk, melyekhez az elmúlt években általunk feldolgozott több, mint 50 szótár szolgál anyaggal. Kitérünk a papírszótárak szerkezeti-tipográfiai egységének és egyértelműségének fontosságára, valamint felvázoljuk, miért jelenthet egyedülálló és eredményes módszert a papíralapú szótárak feldolgozásában, XML-adatbázissá alakításában az általunk kidolgozott elképzelés és a rá épülő alkalmazások.

1 Bevezetés

Cégünk, a MorphoLogic, 1994 óta foglalkozik elektronikus szótárak kiadásával. Kezdetben szószedeteket, és strukturált nyelvi forrásanyagokból előállított, egyszerűsített szerkezetű szótárakat jelentettünk meg, a cégen belül kialakított egyedi formátumban. Szótáraink versenyképességét a szótári tartalom intelligens keresésére[31] alkalmas hozzáadott nyelvi háttér-információkkal és -adatbázisokkal teremtettük meg. Ez a szótárpiacon egyedülálló nyelvtechnológiai módszer jogosan ébresztette azt a reményt, hogy a szótári tartalomhoz hozzáadott nyelvészeti tudás felkelti majd na-

[31] A MorphoLogic szótárprogramjai a kezdetektől támogatják a szótári tartalom nyelvi elemzésen és szóalaktani háttérinformációkon alapuló keresését, a magyar és néhány kelet-európai nyelv mellett többek között angolul és németül is. Az intelligens keresésről illetve definíciójáról bővebben lásd: Prószéky – Kis: Számítógéppel emberi nyelven. Szak Kiadó, Bicske (1999) 187-193; 207-213 p.

gyobb (esetleg nemzetközi) szótárkiadók érdeklődését is, és lehetőségünk lesz szótáraik közös, elektronikus kiadására.

A nemzetközi szótárkészítési módszerek vizsgálata azt mutatta, hogy az elismert szótáríró műhelyek egyre inkább a nyelvi adatbázisok jelölőnyelveken[32] való leilletve átírásával alkotják meg szótáraikat. Így ha együttműködésre törekszünk velük, szótári rendszerünket és szótárkészítési módszereinket olyan irányban érdemes fejlesztenünk, mely lehetővé teszi jelölőnyelvi kódban tárolt nyelvi adatbázisok fogadását és kezelését.

A jelölőnyelvek alkalmazásáról szóló döntés eredményeként szótári rendszereinket és szótárszöveg-feldolgozási módszereinket is át kellett alakítanunk. Döntésünket megkönnyítette, hogy a jelölőnyelvi adatbázisok alkalmazásának számos előnye van:

- a jelölőnyelvvel leírt adatbázisok megkönnyítik a szótári adatbázis egyben tartását, egységes kezelését, karbantartását, az anyag bővítését, fejlesztését;
- a szótári adatbázisok egységes, könnyen reprodukálható szerkezetben (SGML, XML) való tárolása lehetőséget teremt arra, hogy munkánkba más is bekapcsolódhasson;
- az egységesen kódolt adattípusok (jelölőnyelvi címkék, elemek) egyszerűbbé teszik új programfunkciók kialakítását (például új keresési, megjelenítési feladatok megoldását);
- az egységes szerkezettel, illetve annak kialakításával könnyebben felismerhetővé és megoldhatóvá válnak a szerkezeti hibák, tévesztések, hiányok, elírások stb.;
- az egységes szerkezetű jelölőnyelvi adatbázis az elektronikus szótárkiadás mellett papírszótárak szerkesztésére, egységesítésére, kiadására is alkalmas, jól definiált szerkezeti egységeinek egyszerű kezelhetőségével (keresés, legyűjtés, stb.) forrásául szolgálhat különböző nyelvi kutatásoknak;

Az új technológia megteremtette új szótárkészítő- és feldolgozó programok, új módszerek létrehozásának igényét is. Legfontosabbnak egy olyan eszköz megteremtése látszott, mely emberi segítséggel képes egy nem jelölőnyelvben kódolt szótáradatbázist gyorsan, olcsón jelölőnyelvi adatbázissá alakítani, szerkezetileg pontosan reprezentálva a létrejövő dokumentumban az adott szótár szerkezetét. Erre azért volt szükség, mert olyan szótárak feldolgozásának és szótárprogram alá illesztésének lehetőségét is szerettük volna megteremteni, melyek még csak hagyományos, tipografikus formájukban, ún. papírszótárként léteztek, illetve amelyeknek elektronikus formája és szerkezete nem, vagy nem sokban tért el egy tipografizált szótárétól. Az általunk feldolgozásra és megjelentetésre érdemesnek tartott magyarországi szótárak többsége még csak ilyen formátumban létezett.

A jelölőnyelvek alkalmazására való áttérés még egy megoldandó feladatot adott. Meg kellett határoznunk a szótárleíráshoz használt nyelvi formalizmust. Mivel az elektronikusan tárolt szövegek, ezen belül is a szótárszerkezetek leírására már létezett

---

[32] Jelölőnyelv: markup language; számítógépes tartalmak leírására szolgáló ún. tartalomleíró nyelvek, melyek nemzetközi formátumszabványok alapján egységes jelekkel írják le egy adott tartalomról az önmagán túlmutató információkat: a kódolt tartalom típusát adott osztályozási rendszerben (adatszerkezetben), helyét, szerepét egy nagyobb tartalmi egységben, a tartalom formázási információit, stb. A világhálón szereplő elektronikus tartalmak egységes leíró nyelve, a HTML mellett ide tartoznak – többek között – az általunk szótári adatbázisok kódolására alkalmazott SGML (Standard Generalized Markup Language) és XML (eXtensible Markup Language) nyelvek is. Cikkünkben a továbbiakban a jelölőnyelvekre való hivatkozással az általunk alkalmazott SGML és XML nyelvekre utalunk.

jelölőnyelvi ajánlás, elhatároztuk, hogy amennyire ez lehetséges, a szótárszerkezetek kialakításában igazodunk a TEI[33] által kidolgozott javaslatokhoz.

A cikkben néhány, általunk feldolgozott szótárból[34] közlünk részleteket, példákat, illetve hivatkozunk a bennük tapasztaltakra.

2. Kísérletek a szótári szöveg jelölőnyelvi adatbázissá alakítására

2.1 A lineáris feldolgozási elmélet

Egy szótár sajátos szerkezete a szótárt alkotó szócikkekre vonatkozó szabályosságok elemzésével mérhető fel és alkotható meg, ezért a szócikket határoztuk meg leendő feldolgozóprogramunk legnagyobb felismerendő és feldolgozandó egységeként. Ennél nagyobb adatszerkezeti egységeket már nem kellett felismernie a programnak illetve tükröznie a létrehozandó jelölőnyelvi adatbázisnak.

Az elmélet, melyre leendő eszközünk működését felépítettük, annak felismerésére épült, hogy bármely jól szerkesztett papír- illetve tipografikusan leírt elektronikus szótár már önmagában hordozza szerkezetét, melyet a jelölőnyelv alkalmazásával tulajdonképpen csak láthatóvá teszünk. Ezt a szerkezetet a szótár tipográfiai megformálásával (betűtípusok, grafikai jelek, központozás, szövegtagolás stb.), valamint kötött és szabad tartalmainak[35] felismerhető sorrend – rendszer – szerinti szerepeltetésével jeleníti meg.

A szótárak szerkezetét tehát – adott szótár szócikkeire nézve – kötött elemsorrendű struktúrának tekintettük, melyben a tartalommal bíró elemi alkotórészek, egységek sorrendisége határoz meg nagyobb szerkezeti egységeket. Ezek a nagyobb egységek alakítják ki a szótári szócikk általános szerkezetét. Ezt az általános struktúrát kell az embernek leírni úgy, hogy a szerkezet adott pontjaihoz rendelt tipográfiai információk segítségével egy program felépíthesse a kívánt szövegstruktúrát. Feldolgozási elméletünk szerint egy szótár általános szerkezete alkalmas arra, hogy tartalmát a

---

[33] A TEI – Text Encoding Initiative – egy nemzetközi tudományos projekt, melynek célja megteremteni egy egységes elektronikus kódolási rendszert összetett szerkezetű szövegstruktúrák szerkezetének interpretálására. Ennek az átfogó munkának egyik fejezete a szótári struktúrák elektronikus kódolására ad ajánlásokat. A TEI-ről bővebben a http://www.tei-c.org, a szótári ajánlásokról a http://www.tei-c.org/P4X/DI.html webcímen olvashatunk.

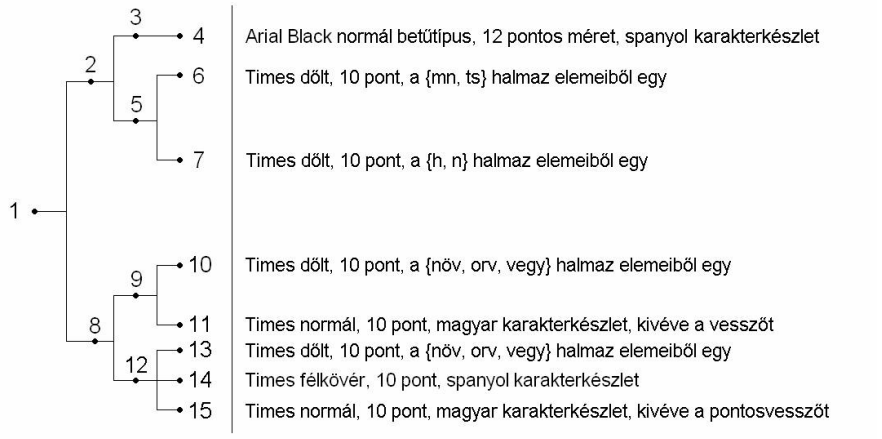
[34] Pálfy Miklós: Francia-magyar kéziszótár. Grimm Kiadó, Szeged; első kiadás: 1999
Hessky Regina: Német-magyar kéziszótár. Nemzeti Tankönyvkiadó–Grimm Kiadó; 1. kiad.: 2000
Forgács Tamás: Magyar szólások és közmondások szótára. Tinta Kiadó, Budapest, 2003
Dorogman György: Spanyol-magyar kéziszótár. Grimm Kiadó, Szeged; 2., jav. kiadás: 2002
Német-magyar Üzleti Nagyszótár (Hamblock/Wessels/Futász). Tudex Kiadó, 2001-2004
Angol-magyar és Magyar-angol Nagyszótárak. Akadémiai Kiadó, Budapest, 1998
Német-magyar és Magyar-német Nagyszótárak. Akadémiai Kiadó, Budapest, 1998

[35] Kötött tartalmúnak nevezem egy szótárszöveg szerkezeti elemei közül azt, melynek lehetséges konkrét, teljes tartalmai, vagy a teljes tartalmat lefedő résztartalmai egy jól definiálható, zárt karakterlánc-halmaz elemeiként meghatározhatóak, ilyen például a szófajkészlet, a nyelvtani nem jelölése, vagy a szakjelzetek. Szabad tartalmú az elem, ha tartalmaira konkrétan nem, csak karakterosztályokkal, karakterkészletekkel hivatkozhatunk.

szócikk első tartalommal bíró elemétől – általában a címszótól – kezdve, elemről elemre, sorrendben, kihagyások nélkül feldolgozhassuk úgy, hogy közben az elemek és elemhatárok tipográfiai kódjait használjuk fel a szerkezetkialakító program vezérlésére[36]. Az egyes szerkezeti elemekhez rendelt, előre felmért és definiált tipográfiai tulajdonságokat az elemzendő szövegrész tulajdonságaival összevetve ismeri fel a program, hogy a szótári szócikk elemzésének adott pontján milyen elemek kialakítására van lehetősége.

Rendszerünk adatleíró modelljét a következő, tipografizált szócikkek alatti faszerkezetben ábrázoljuk, feltüntetve, hogy elemző és szerkezetkialakító rendszerünknek milyen információkat kell ebből a szövegből kinyernie és tárolnia:

cloro *h, vegy* klór
clorofila *n, növ* klorofill, levélzöld
coactivo *mn* kényszerítő
coartar *ts* akadályoz, korlátoz
cobalto *h, vegy* kobalt; **azul de ~** kobaltkék; *orv* **bomba de ~** kobaltágyú



1. szócikk	5. nyelvtani csoport	9. jelentéscsoport	13. szakjelzet
2. szócikkfej	6. szófaj	10. szakjelzet	14. kifejezés
3. címszóforma	7. nyelvtani nem	11. jelentés	15. kifejezésjelentés
4. címszó	8. szócikktest	12. kifejezéscsoport	

1.ábra. A példaszócikkek mindegyikére érvényes szótári adatszerkezet fastruktúrában ábrázolva, a levélszintű elemekhez tartozó, a rendszer által tárolandó információkkal

A szótári szócikk szerkezetét megjelenítő faszerkezetben a szövegtartalom a fastruktúra legutolsó elemein, levelein helyezkedik el. A csomópontok képviselik a kisebb, hasonló szerepű egységeket átfogó szerkezeti elemeket. A szótár tagolóelemeit – a központozást, grafikus szimbólumokat, jeleket stb. – a fabejárás döntési pontjainak, elágazásainak (ÉS/VAGY ágak, elemismétlődések, opcionális

[36] Erre a szótári tartalmakat sorrendben, kihagyások nélkül feldolgozó módszerre hivatkozom lineáris feldolgozási elméletként.

elemek) reprezentánsaként képzeltük el. Ezek jelölik a szótárban az egyes szerkezeti egységek közötti váltást. Tehát a fa elemei (a levelek és az elágazások, melyek sokszor reprezentálnak csoporthatárokat is) modellünkben nemcsak az adott, konkrét egyedi tartalmakat kell hordozzák, hanem minden egyéb olyan általános információt, szabályosságot is, mely a szótárban megjelenített szövegrészek sajátja. Például jól látható, hogy bár a fában minden lehetséges szerkezeti elemet ábrázoltunk, a levélelemek nem mindegyike kötelező. Ennek az információnak is meg kell jelennie a rendszerben. Erre elképzelésünk szerint külön szerkezetleíró állomány szolgált.

Első lépésben a részletes szövegvizsgálat után ebben a szerkezetleíró fájlban határozható meg a dokumentum általános szerkezete. Ezután minden levélelemhez meg kell határozni a szótárban hozzá rendelt tipográfiai és tartalmi információkat – a betűtípust, a kötött tartalmakat, a karakterkészletet vagy karakterosztályokat, illetve bizonyos, az elem tartalmából kizárható karaktereket, karaktercsoportokat. A szerkezetleírásban kell meghatározni a különböző szövegelemek szomszédsági viszonyainak szabályosságait is, például a központozás, szimbólumkészlet azon elemeit, melyek két vagy több elem illetve elemtípus határán kötelezőek.

A szótárt ezek után a szócikkszerkezethez kialakított faszerkezetben fentről lefelé haladva kell feldolgozni úgy, hogy a szerkezetelemzés a fa minden levelét érintse. Az elemző pedig a szerkezet adott helyén eldönti, a szöveg következő feldolgozandó része megfelel-e a szerkezetben soron következő lehetséges elem (levél), illetve szomszédsági viszony formai követelményeinek[37]. Olyan eszközt kellett találnunk, mely képes felismerni, azonosítani egy szöveg tipográfiai megjelenését, és azt képes a tartalommal együtt tárolni. Ezután egy szövegelemzési és szerkezetkialakítási szakaszban a szerkezeti elemekhez előre definiált tipográfiai feltételek, valamint a feldolgozandó szöveg tartalmi tulajdonságai alapján az elemzendő szövegen végigfut, és megalkotja a kívánt szerkezetű jelölőnyelvi adatbázist. Mindezt úgy, hogy lehetőségünk legyen a munka bármely fázisában az előre definiált szerkezet, a tipográfiai feltételrendszer vagy az elemzendő dokumentum módosítására.

2.2 A Yacc-Lex programpáros

Erre a célra megfelelőnek látszott egy tipográfiakivonatoló segédprogram és a Yacc-Lex programpáros együttese. Első lépésben a teljes tipografikus adatállományból egy olyan dokumentum készült, melyben .txt formátumban együtt szerepeltek a tartalmak és a tartalmakhoz tartozó stílusinformációk, tipográfiai tulajdonságok is. Így lehetővé vált, hogy a csak szövegformátumú állományokon dolgozó Lex és Yacc számára feldolgozható anyagot szolgáltassunk.

A Lex program számára a szótár tartalmát karakterosztály-definíciókkal, illetve kötött tartalmú szerkezeti elemek meghatározásával ún. tokenekre bontottuk, melyek alapján a Yacc programmal és saját kódrészletekkel elértük, hogy a létrejövő C++ programkód adott szerkezetű XML-lé kódolja a szövegben a neki megfelelő formalizmusban szereplő szócikkeket. Hamar kiderült azonban, hogy ezzel a módszerrel sokszor nehézkes, sőt, néha további szövegátalakítás nélkül lehetetlen XML-

37 Más szavakkal: az egyes tartalmas elemek azonosított, egy adott elemre nézve elméletben egységes tipográfiai, sorrendi, szomszédsági, opcionalitási jellemzőit információként használjuk arra, hogy egy dokumentumot feltérképező program a szótárszerkezet felderítésében éppen adott szócikk mely elemének felismerésénél tart.

adatbázis létrehozása. A szerkezetelemző a feldolgozandó szövegegységben (szócikkben, entryben) ugyanis mindig csak a feldolgozási folyamatban éppen soron következő feldolgozandó elemi szövegegységet látja. Ha a dokumentum faszerkezetében az elemző döntési ponthoz jut, ott a soron következő szövegegységet (karaktereket) mindaddig feldolgozza, míg annak tulajdonságai megfelelnek a faszerkezetben egy adott döntési pont után levélszinten várt elem tartalmi és formai megkötéseinek. Arra azonban képtelen, hogy az elemzendő szövegrész környezetéből további következtetéseket vonjon le. Egy ilyen elemzőt pusztán a szótárban eredendően meglévő tipográfiai és tartalmi megkötések alaposan félrevezethetik. Gondoljunk csak egy szótár sokszor igen bonyolult szerkezetére, az azonos tipográfiájú, de eltérő tartalomtípusú elemekre, melyek a szótár legkülönbözőbb helyein megjelenhetnek. A különböző nagyobb szerkezeti egységekben megjelenő ismétlődő elemek illetve részszerkezetek ugyanígy a programozott szerkezetértelmezés többértelműségének forrásai lehetnek. Adott esetben olyan szócikkszerkezet is létrejöhet az ilyen, csak a következő elemzendő egységet látó és feldolgozó elemzővel, mely megfelel ugyan az elemzési sorrendben várt összes tipográfiai információnak, a szerkezet mégis hibás lesz: nem a szótár valódi tartalmát és szerkezeti egységeit tükrözi. Az eszköz legfőbb korlátja az volt, hogy az elemzés azonnal megakadt, amint a soron következő adatsor nem illeszkedett ahhoz a formai feltételrendszerhez, melyet az elemző az elemzési fában általa éppen bejárt helyen várt. Ilyenkor a program már nem próbálkozott a szerkezet más módon történő kialakításával.

A problémára megoldást kínált a szótár *egészét* ellátni vezérlőkarakterekkel, mégpedig *minden* olyan döntési ponton, ahol a tartalmi és tipográfiai feltételek önmagukban nem voltak elegendőek az elemző kiszolgálásához. De ez igen nehézkessé és lassúvá tette a munkát. Olyan rendszerre volt szükségünk, mely a lehető legkevesebb emberi előfeldolgozással képes egy szótár jelölőnyelvi adatbázissá alakítására.

Egy új szerkezetelemző és -kialakító program megalkotásának első lépéseként összefoglaltuk a YACC-LEX rendszer számunkra meghatározó korlátait:

- csak a következő feldolgozandó tipográfiai egységig lát előre: amennyiben az megfelel a várt formai követelményeknek, azonnal és véglegesen létrejön a következő szerkezeti elem; minden egyéb esetben elakad az elemzés
- nem képes egy kérdéses elemzési ponton annak környezetéből nyert információk alapján dönteni, hogy adott ponton választható több szerkezeti elem melyikével folytassa a jelölőnyelvi adatbázis szerkezetének felépítését
- sikeresen elemzettnek minősíthet szócikkeket úgy is, hogy az adatbázis adott egysége (egy szócikk) a megadott tipográfiai szerkezetleírás szerint többértelmű, és a program nem a helyes szerkezetet választotta; a program ilyenkor *nem jelzi* a szerkezeti többértelműséget
- csak szöveges bemenetet tud fogadni; így speciális karakterek illetve kevert kódlapok esetében nincs biztosítva a tartalom egységes átkódolása – esetlegesen tartalomváltozást, tartalomvesztést okozva

Ezekre a hiányosságokra kellett egy létrehozandó új programnak megoldást találnia.

2.3. A MarkUpWizard, egy új szerkezetelemző és -kialakító program

Az előző fejezetben vázolt problémákra a cégen belül fejlesztett MarkUpWizard (jelölőnyelv-varázsló) program[38] jelentett megoldást. A program legfontosabb adottsága, hogy adott formális nyelvtan (előre definiált dokumentumszerkezet-leírás, valamint tipográfiai és tartalmi feltételek) szabályainak eleget tevő RTF dokumentumot képes XML dokumentummá alakítani. (Ez azt is jelenti, hogy képes minden olyan adatállományt feldolgozni, melynek formátuma tartalmi és formai információvesztés nélkül RTF-fé konvertálható.)

Az alkalmazás szükségtelenné tette tipográfiakivonatoló program használatát. A felhasználónak két feladata van: a szótár szerkezetét leíró formális nyelvtan elkészítése, és a szerkezeti elemekhez rendelt tartalmi és tipográfiai tulajdonságok leírása. Ezek elkészítésével el lehet kezdeni a feldolgozandó dokumentum átalakítását. A program UNICODE karakterkódolás használatával küszöböli ki a nem kívánt adatvesztést, és adatváltozást.

Az elemzés eredménye első próbálkozásra szinte biztos, hogy nem tökéletes. Elég, ha csak a szótárakban jelen levő formai, sorrendi stb. hibákra gondolunk. A munka elején azonban a felhasználó legtöbbször még nem méri fel és írja le tökéletesen a szótár szerkezetét. Így az elemzés eredményétől függően a nyelvtan és/vagy az RTF dokumentum módosítása után a második vagy mindkét lépést meg kell ismételni, egészen a kívánt eredmény eléréséig. (Tehát az elemzési eredmények hatással vannak a szótári tartalomra.) További fontos tulajdonsága a programnak, hogy egy szócikk elemzése közben az éppen elemzésre következő tipografikus adatból kialakított szerkezeti elemről nem dönti el azonnal, vajon az megengedhető-e a szótár szerkezetében. Amikor az elemző a feldolgozandó bemenetben olyan döntési ponthoz jut, melynél az elemzés folytatását többféleképpen[39], vagyis egymástól eltérő szerkezeti egységek kialakításával is folytathatja, kiválasztja az egyik lehetőséget, és a dokumentumban sorrendben később következő elemek tipográfiai információi igazolják vagy cáfolják a választás helyességét. Amennyiben a több lehetőség közül választott szerkezeti elem után következő adatok nem igazolják a választást és az elemzés elakad, az elemző visszalép a kérdéses pontig, és adott ponton megengedhető, addig még nem próbált szerkezeti elem létrehozásával kísérli meg érvényes szerkezet létrehozását. Amennyiben több ilyen módon megalkotható teljes szócikkszerkezet létezik, az eredményfájlban a szerkezet többértelműségére megjegyzés figyelmeztet. Ha csak egy, a kívánt formai követelményeknek eleget tevő teljes elemzés születik, az helyesnek minősül. Amennyiben az elemzés sikertelen, a program a leghosszabb, még elemezhető részfát XML-ben az eredményfájlba írja, a nem elemzett részeket pedig szabványos jelölőnyelvi megjegyzésben szerepelteti. A lezáratlan, de megkezdett elemeket a program lezárja. Az elemzési eredményeket mindvégig egy állományban tartja, ugyanabban a fájlban épülnek fel a helyes szerkezetű szócikkek, ugyanitt jelennek meg a hibaüzenetek és a többértelműségre utaló megjegyzések is. Mindezt

---

[38] A programot Pál Miklós, cégünk egyik vezető fejlesztője írta

39 Általában egy szótári szócikk szerkezetében számos olyan hely található, ahol a szerkezet vagylagos struktúrákat, elemeket, elemcsoportokat enged meg. Adott bemenő adatsorban az elemzés elérhet olyan pontig, melyen az elemzésre következő adat tipográfiai jellemzői többféle, adott elemzési helyen kialakítható szerkezeti elem tipográfiai kívánalmainak is eleget tesznek. Ezt nevezzük döntési pontnak; ilyenkor egy elemzőnek választania kell, milyen elem kialakításával építi tovább a kialakítandó szerkezetet.

úgy teszi meg, hogy a létrejövő adatok szintaktikailag mindvégig kielégítik az XML-szabványt, lehetővé téve, hogy az állománnyal XML-kezelő eszközökkel is dolgozhassunk.

3 Tapasztalatok a szótár jelölőnyelvi adatbázissá alakításáig

3.1 A jelölőnyelvvé alakítás hasznai - egységes szöveg; hiányok, hibák feltárása

Ahogy az előzőekből kiderült, a szerkezetelemző program a felmért szerkezet egyre pontosabb leírásával, illetve a kiinduló dokumentum hibás részeinek javításával közelít a végleges adatbázis kialakítása felé. Mivel a szerkezet leírása igen szigorúan igazodik a szótár valódi tartalmi, szerkezeti struktúrájához, ezért a jelölőnyelvi adatbázis kialakítása közben rengeteg hibára, hiányosságra, figyelmetlenségre fény derül. A szerkezetleíró állományban pontosan definiált kötelező és opcionális elemeket, a kötött tartalmakat, az elemek meghatározott sorrendjét, a szöveg grafikai tagolását, az adott szerkezeti elemekre vonatkozó pontosan megadott tipográfiai tulajdonságokat a program a dokumentum egészén számon kéri. Éppen ezért ez a szövegfeldolgozó rendszer nagyon jól használható a szöveg egységének, tartalmi helyességének javítására is. Az elemzés közben kapott eredmények rámutatnak a hiányokra, hibákra, hiszen az elemzés akkor akad el, amikor az ember által leírt és a programon keresztül megkövetelt formai, szerkezeti feltételeknek a dokumentum nem felel meg. Így az elemzés visszahat a dokumentumra, javítva formai, tartalmi egységét, felhívva a figyelmet az át nem gondolt szerkezetre, hibákra, tévesztésekre.

Általános tapasztalatunk a szövegek feldolgozásánál az, hogy hibátlan szótár nincs! Amelyik szótárban a munka egyötödének ráfordításával a szócikkek 75 százaléka szerkezetileg helyesen létrejön, az már egy átgondolt, nagyon jól szerkesztett szótár! A fennmaradó részben találhatók azok a szerkezeti sajátosságok, problémák, hibák, melyeket további szerkezetmódosítással, vagy a kiinduló dokumentum javításával kell megoldani.

3.2 Az átalakítás tapasztalatai – hibák, megoldási javaslatok

A szótárakban többféle típushibával találkozunk, melyek nehezítik a jelölőnyelvi dokumentum létrehozását. Előfordul például elemek sorrendjének felcserélődése, a szótárban elvileg meglévő kötött tartalmak tartalmi variálódása, kötelező tipográfiai tulajdonságok átalakulása vagy elveszése, a szövegtagolás megszokott formalizmustól való eltérése. Ezek a legtöbb szótárban előfordulnak.

Annak érdekében, hogy ezek a hibák ne forduljanak elő a szövegben, előre definiálni kell a szótár szerkezetét. Érdemes próbaszócikkekkel kialakítani egy olyan, szigorú szerkezetet, melyet aztán minden szótárszerkesztő leírva kézhez kap, a kötelező elemtartalmak (szófajok, földrajzi használati körök, szakjelzetek stb.) pontos halmazelem-leírásával, az elemek kötelezőségének vagy elhagyhatóságának gondos feltüntetésével. Egy XML-szerkesztővel írt szótárban elvileg a program képes figyelni számos ilyen információra. Mégis, tapasztalatunk az, hogy egyrészt a mai szótáríró gene-

ráció még nem szívesen használ ilyen programokat, vagy, ha használ is, sokszor nem veszi figyelembe a szerkezeti elemek valós szótárbeli szerepét. Megelégszik azzal, ha talál egy olyan szerkezeti elemet, melyre nézve a tipográfiai megszorítások megegyeznek azzal a formai megjelenéssel, melyet az adott leírandó információnak elképzelt. Mivel ezek az elemek legtöbbször az elhagyható – és általában nem kötött tartalmú – elemek kategóriájából kerülnek ki, nagy esélye van különböző szerepű, de azonos tipográfiájú elemtartalmak keveredésének. Ez a keveredés azonban elkerülhető, ha a szótárszerkesztők a különböző szótári tartalmakat más-más tipográfiával jelenítik meg. Két-három fonttípus helyett érdemes négy-öt, egymáshoz illeszkedő, mégis markánsan más fontot használni.

Könnyebb a szótári szöveg egységének megőrzése, ha jól látható grafikus szimbólumok, jelek választják el egymástól a nagyobb szerkezeti egységeket. Ráadásul ennek kettős előnye van: segíti a szótár áttekinthetőségét, segít eligazodni a szótárban, és a szöveg feldolgozhatóságát is gyorsabbá, egyszerűbbé teszi. Természetesen lehet egy szótár áttekinthető a különböző grafikai elemek, sematikus ábrák, szövegegységeket tagoló szimbólumok alkalmazása nélkül is. Sokat segíthet a betűtípusok variálása, gondos megválasztása, a szöveg megfelelő tagolása központozással, új sorban kezdett nagyobb szerkezeti egységekkel, az egymástól elkülönítendő sorok szövegbehúzási tulajdonságainak megváltoztatásával. Véleményünk szerint mégis hasznos, ha a szöveg áttekinthetőségét grafikai elemek bevezetésével segítjük.

Annak igazolására, hogy legalább tartalmi és szerkezeti ellenőrzésre érdemes igénybe venni szerkezetelemzőnket, álljon itt néhány feldolgozott szótárból vett példa. Ezekkel illusztráljuk, hányféle hiba érhető tetten még azokban a szótárakban is, melyeket jól szerkesztetteknek mondunk, sőt, azokban is, melyek elvileg már XML-szerkezetben vannak.

Volt olyan szótárunk, melyet két külön formátumból kellett egységes alakra hoznunk. Ennek egyik részét egy szoftvercég kódolta jelölőnyelvi adatbázissá. Az anyag valóban megfelelt a jelölőnyelvi adatbázisokkal szemben felállított formai követelményeknek. Azonban az általánosabb szerkezeti egységek kialakításán túl, a bonyolultabb, és szerkezeti következetlenségeket is tartalmazó anyag feldolgozásánál már nem vették a fáradságot arra, hogy definiálják az elemek sorrendjét, és azt, hogy a szerkezetben kötelezőek-e vagy sem. Így fordulhatott elő, hogy több száz (!) helyen hiányzott a forrásnyelvi kifejezések magyar fordítása, amit a szerzővel, több heti munkával utólag kellett pótolnunk.

Íme még egy fontos tapasztalat: nem mind jelölőnyelvi adatbázis az, ami annak látszik! Egy adatbázisnak nem csak formai követelményeknek kell eleget tennie, pontosan tükröznie kell az eredeti dokumentum tartalmi és szerkezeti sajátosságait is! Mivel feldolgozási módszerünk egyik alapkövetelménye, hogy a szótárban fellelhető szerkezetet olyan szigorú szerkezetben írjuk le, amennyire az magának a szerzőnek koncepciója volt, így nálunk nem fordulhat elő, hogy ezek a hibák rejtve maradjanak.

Találkoztunk még olyan hibákkal, mint például a kötelező elemek kitöltetlenül hagyása – ez legtöbbször szófajok esetében fordult elő. Ha egy szerkezetellenőrző programban jól definiáltak a kötelező tartalmak, minden ilyen jellegű hibára fény derül. Általában véletlen hibák ezek, elfelejtődnek a kitöltendő szerkezetek.

Sokszor fordul elő egymás melletti – főleg minősítési és körülírásra szolgáló – elemek sorrendi következetlensége. Jellemző, hogy amikor fel lehetne állítani sorrendi precedenciákat, akkor sem teszik a szerkesztők ezt meg. Főleg a különböző minősítések, használati rétegek – pl. földrajzi használat, szakjelzetek, stílusrétegek –

keverednek. Nagyon sok szótár nem is tesz tipográfiai különbséget az ilyen adatok között. A feldolgozott szótárak szolgálnak megoldással is: megkülönböztethetjük az adatokat, ha például a szakjelzetek nagybetűvel kezdődnek. A minősítések jelölésére szolgálhat az adatok zárójelben (), < >, való szerepeltetése. Ezek csak ötletek. A lényeg: egymástól jól megkülönböztethető, lehetőleg minél kevesebb célra használt jelek, tipográfiai tulajdonságok alkalmazása.

Elgondolkodtató a szótárakban elvileg kötött tartalmak variálódása is. Volt olyan szótárunk, ahol a tárgyas ige megadására a következő elemek szolgáltak: *i ts, ts i, tsi, ts*. Ugyanebben a szótárban találunk példát a tartalmi ellenőrizetlenség súlyosabb esetére is. A szótárban szerepelő **lótenyésztő** szó szófajaként a *ló* volt megjelölve. Szintén a tartalmi ellenőrizetlenség példája a következő: Ha a hozzánk érkezett elektronikus anyag változatlan formában jelent volna meg papíron, a **szovhoz** szó szófaja 'szocialista realista főnév' (*fnszoc.r.*) maradt volna. Jó példa ez azonos tipográfiai tulajdonságú, eltérő szerepű elemek 'összeolvadására'. A papírváltozatban természetesen már a helyes, *... fn; szoc.r. ...* szerkezetű adat szerepelt. Ez utóbbi két példát egy – elvileg – XML-szerkezetű szótárból gyűjtöttük. Látható, hogy még az ilyen, szerkesztettnek mondott szótáraknál is összekeveredhetnek tartalmak, pusztán tipográfiájuk hasonlósága vagy azonossága miatt. Ez is bizonyítja, hogy mennyire fontos a kötött tartalmak valódi ellenőrzése. Természetesen előfordul olyan eset is, hogy a tartalom helyes, de nem a neki megfelelő tipográfiai megjelenítést kapja. Mivel elemzőnk egyszerre értékel formai információkat, és a szövegegységek kialakítandó szerkezetben elfoglalt helyét is, az ilyen hibák sem maradnak rejtve.

A szerkezet kialakítása közben is érdekes dolgokat tapasztalhatunk. A következő sematikus szócikkrészlet példa a felesleges információismétlésre, megmutatva, sokszor mennyire nem átgondolt egy szótári struktúra:

<címszó> *mn*, **<címszóvariáns>** mn ...

Valóban nincs értelme *egy* szócikkben szerepeltetett több címszóvariáns mindegyikéről elmondani, hogy melléknév. Nem tartjuk szükségesnek egyedi és típushibák további felsorolását. Célunk az volt, hogy a fenti néhány példa egyértelművé tegye: a szótárak szerkezeti és tartalmi egységességének kialakításához szükség van mind szerkezeti, mind tartalmi ellenőrzésre.

Végezetül álljon itt még egy általános tapasztalat. A szerkezetkialakító munka vége felé sokszor találkoztunk a következő problémával: a néhány, még nem elemzett szócikk szerkezete megkövetelheti újabb elemek szerkezetbe vételét. Néha valóban szükség van a szótárstruktúra leírásának bővítésére, de legtöbbször az ilyen 'új' szerkezetelemek bevezetése felesleges. Nagyon valószínű, hogy inkább a szótár átgondolatlanságáról van szó, és a közölni kívánt dolgokat le lehetne írni más, a struktúrában már meglévő elemmel. Előfordul persze ellenpélda, de ha egy elem csak öt-tíz, vagy kevesebb esetben fordul elő, érdemes gyanakodni arra, hogy a szerző másként, egységesebben, egyszerűbben is leírhatta volna ugyanazt.

4 A nyelvi egyértelműsítés, előkészítés az intelligens kereshetőségre

4.1 A nyelvi egyértelműsítés szerepe, fontossága

A kívánt szerkezetű XML létrejötte után kezdődik az a munkaszakasz, amelyik már a kerestetni kívánt szövegtartalmak intelligens megtalálásáért felelős. Tudjuk, hogy a papírszótárak, főleg helytakarékosság és a fölösleges ismétlések elkerülése érdekében élnek a szövegbehelyettesítés (tilde), szövegrövidítés (perjelek, per-kötőjelek), jelentésrész-összevonás (perjelek, elhagyható zárójelek) módszerével. Míg azonban az emberi olvasásra szánt ilyen jellegű szótárakban a felhasználó – általában – intuitív módon be tudja helyettesíteni az adott, hiányzó szövegrész(eke)t, értelmezni tudja a sűrített, rövidített tartalmakat, addig a számítógép önmaga nem képes értelmezni a szótárakban ilyen megoldással előforduló tartalmak összetartozó egységeit. Így, előzetes értelmezés, segítség nélkül nem is lenne képes arra, hogy valóban kereshessen az ilyen tartalmakban. Ezt a jelentésegyértelműsítést, jelentésfeloldást, jelentésszöveg-dekódolást végzi el az adatbázison a nyelvi normalizálás munkafázisa.

4.2 A nyelvi egyértelműsítés munkafázisai, megoldandó feladatai

A nyelvi egyértelműsítés első szakaszában, a normalizálásban ezeket a szöveget rövidítő jeleket oldjuk fel, és értelmezzük. Az összetartozó szövegelemeket egymáshoz illesztjük, és a már egységbe fogott, egyértelműsített szövegegységeket a dokumentumban nekik megfelelő, rövidített szövegegységek mellé illesztjük. A szövegkeresések a továbbiakban az egyértelműsített szövegekhez fordulnak, míg a szövegek megjelenítésére továbbra is az eredeti szövegek szolgálnak.

Az intelligens keresést segítő következő munkafázis már a keresendő szótári tartalom nyelvi háttéradatokkal, szótövekkel való kiegészítése. Ezt a munkaszakaszt azonban, mivel felmerülő nehézségei, problémái nem szótárszerkesztési problémák, a következő, tapasztalatokat, javaslatokat tárgyaló részben már nem érintjük.

4.3 Hibák, tapasztalatok – hogyan tud segíteni a szótáríró?

Minden szövegrövidítési típusra adhatók olyan általános szabályok, melyeket általában a rövidített szövegrészek többségénél alkalmazni lehet az egyértelműsítésre. A tilde karakter feloldásakor például általában a címszó a rövidített, behelyettesítendő tartalom. Volt azonban olyan szótár, melyben egy szócikken belül sem volt egységes az alkalmazása: hol a teljes szócikket kellett behelyettesíteni, hol a ragozott alakokhoz tartozó megrövidült tőallomorfot. A szövegegyértelműsítést a jelölési szokások szócikkenkénti különbözősége nehezítette. Tehát jól át kell gondolnunk a szövegrövidítés alkalmazott módjait. Ha nem egységesen használjuk ezeket, automatikusan nem lehet a teljes tartalmat reprodukálni. Márpedig egy szótárfeldolgozó rendszerben ez lenne a cél.

Az alapelvek a szövegrövidítés bármely formájával kapcsolatban ugyanazok: úgy használjuk őket, hogy a szótár bármely pontján ugyanazt a szövegegység-

visszaállítási algoritmust alkalmazva minden azonos típusú feloldandó tartalom egyformán, helyesen egyértelműsítve hozza létre a teljes egybetartozó szótári tartalmat. A továbbiakban bemutatjuk azt a szövegrövidítési eljárást, melynél az egyértelműsítést segítő egységes szövegkódolásra megoldást is tudunk ajánlani.

4.4. A perjelek használata – javaslat összetartozó szövegek kódolására

A perjel a szövegben felcserélhető tartalmak jelölésére szolgál. Általában két, egymással a jelentésben felcserélhető szót választ el egymástól. Sokszor azonban a perjelek két oldalán szereplő alternatívák nem csak egy-egy szót, hanem egész szövegrészeket fednek le. A számítógép maga nem tudja ezeket a szokásostól eltérő szerkezeteket feloldani. Álljon itt egy példa: *nem szavazó/szavazásra nem jogosító részvény.* Vajon értené-e egy a magyar nyelvvel csak ismerkedő szótárolvasó, mit jelent a *nem szavazó nem jogosító részvény,* vagy a *nem szavazásra nem jogosító részvény*? A problémára több megoldás is létezik. El is kerülhetjük az ilyen szerkezetek leírását, de segíthetjük is a szótárolvasót, amint az a következő, nyomtatásban megjelent szótári szócikkrészletben is látszik:

{arcára fagy/ráfagy az arcára/lefagy az arcáról} a mosoly arcának (...) érzelmi megnyilvánulása hirtelen megszűnik ...

Itt az összetartozó tartalmak egyértelműen egymáshoz rendelődnek. Nemcsak a szöveg feldolgozását segíti az ilyen megoldás, de az olvasó is könnyebben eligazodik a szövegben. Természetesen fenti példa csak javaslat, több, általunk is látott megoldás lehetséges az összetartozó szövegek egymáshoz rendelésére.

A nyelvi egyértelműsítés a megfelelő szabályok megalkotásával és betartásával egyszerű, sablonok mentén végezhető munka. Minden szövegrövidítési eljáráshoz megtalálható egységes szövegösszerendelő formalizmus, egyértelműsítési filozófia. Ezek közül néhányat tartalmilag is megjeleníthetünk a feldolgozandó szövegben, segítve a gépnek a szótárszöveg feldolgozásában. Azokat, melyek a szótárfelhasználó számára is egyértelműbbé, átláthatóbbá, érthetőbbé teszik a szótárszöveget, érdemes a szótár felhasználóknak szánt formájában is szerepeltetni.

5. Összegzés

Cikkünk összegzéseként elmondhatjuk, hogy a szótáradatbázisok minőségén, tartalmi egységességén sokat javít szerkezetelemzési módszerünk, legyen az tipografizált formában írt, vagy XML-szerkezetben lévő szótár. A feldolgozás eredményeként a szótár helyes szerkezetben, egységesen, áttekinthetően kerül a felhasználók kezébe. A kialakított jelölőnyelvi adatbázissal pedig kihasználhatók mindazok az előnyök, melyeket cikkünk elején már felsoroltunk.